Construction and Querying of Large-scale Knowledge Bases

Part II: Schema-agnostic Knowledge Base Querying
Transformation in Information Search

Desktop search

Mobile search

Lengthy Documents? Direct Answers!

Which hotel has a roller coaster in Las Vegas?

Answer: New York-New York hotel

Surge of mobile Internet use in China

Source: China Internet Network Information Center
People, Places, and Things

Facebook’s knowledge graph (entity graph) stores as entities the users, places, pages and other objects within the Facebook.

Connecting

The connections between the entities indicate the type of relationship between them, such as friend, following, photo, check-in, etc.
QA Engine instead of Search Engine

• Behind the scene: A knowledge graph with millions of entities and billions of facts
Structured Query: RDF + SPARQL

Triples in an RDF graph

<table>
<thead>
<tr>
<th>Subject</th>
<th>Predicate</th>
<th>Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>Barack_Obama</td>
<td>parentOf</td>
<td>Malia_Obama</td>
</tr>
<tr>
<td>Barack_Obama</td>
<td>parentOf</td>
<td>Natasha_Obama</td>
</tr>
<tr>
<td>Barack_Obama</td>
<td>spouse</td>
<td>Michelle_Obama</td>
</tr>
<tr>
<td>Barack_Obama_Sr.</td>
<td>parentOf</td>
<td>Barack_Obama</td>
</tr>
</tbody>
</table>

SPARQL query

```
SELECT ?x WHERE {
  Barack_Obama_Sr. parentOf ?y .
  ?y parentOf ?x .
}
```

Answer

```
<Malia_Obama>
<Natasha_Obama>
```
Why Structured Query Falls Short?

<table>
<thead>
<tr>
<th>Knowledge Base</th>
<th># Entities</th>
<th># Triples</th>
<th># Classes</th>
<th># Relations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Freebase</td>
<td>45M</td>
<td>3B</td>
<td>53K</td>
<td>35K</td>
</tr>
<tr>
<td>DBpedia</td>
<td>6.6M</td>
<td>13B</td>
<td>760</td>
<td>2.8K</td>
</tr>
<tr>
<td>Google Knowledge Graph*</td>
<td>570M</td>
<td>18B</td>
<td>1.5K</td>
<td>35K</td>
</tr>
<tr>
<td>YAGO</td>
<td>10M</td>
<td>120M</td>
<td>350K</td>
<td>100</td>
</tr>
<tr>
<td>Knowledge Vault</td>
<td>45M</td>
<td>1.6B</td>
<td>1.1K</td>
<td>4.5K</td>
</tr>
</tbody>
</table>

* as of 2014

- It’s more than large: High heterogeneity of KBs
- *If it’s hard to write SQL on simple relational tables, it’s only harder to write SPARQL on large knowledge bases*
  - Even harder on automatically constructed KBs with a massive, loosely-defined schema
Certainly, You Do Not Want to Write This!

“find all patients diagnosed with eye tumor”

```
WITH Traversed (cls, syn) AS (  
  (SELECT R.cls, R.syn  
   FROM XMLTABLE ('Document("Thesaurus.xml")  
     /terminology/conceptDef/properties  
     [property/name/text()="Synonym" and  
     property/value/text()="Eye Tumor"]  
     /property[name/text()="Synonym"]/value'  
     COLUMNS  
     cls CHAR(64) PATH './parent::*/parent::*  
                     /parent::*/name',  
     tgt CHAR(64) PATH'.') AS R)  
UNION ALL  
(SELECT CH.cls, CH.syn  
 FROM Traversed PR,  
 XMLTABLE ('Document("Thesaurus.xml")  
 /terminology/conceptDef/definingConcepts/  
 concept./text()=$parent]/parent::*$/parent::*/  
 properties/property[name/text()="Synonym"]/value'  
 PASSING PR.cls AS "parent"  
 COLUMNS  
 cls CHAR(64) PATH './parent::*/  
             parent::*/parent::*/name',  
 syn CHAR(64) PATH'.') AS CH))  
SELECT DISTINCT V.*  
FROM Visit V  
WHERE V.diagnosis IN  
(SELECT DISTINCT syn FROM Traversed)
```

“Semantic queries by example”, Lipyeow Lim et al., EDBT 2014
Schema-agnostic KB Querying

"Barack Obama Sr. grandchildren"

**Keyword query:** query like search engine

"Who are Barack Obama Sr.'s grandchildren?"

**Natural language query:** like asking a friend

<Barack Obama Sr., Malia Obama>

**Graph query:** add a little structure

**Query by example:** Just show me examples
“Find a professor, ~70 yrs., who works in Toronto and joined Google recently.”

Search intent

Graph query

A match (result)
## Mismatch between Knowledge Base and Query

<table>
<thead>
<tr>
<th>Knowledge Base</th>
<th>Query</th>
</tr>
</thead>
<tbody>
<tr>
<td>“University of Washington”</td>
<td>“UW”</td>
</tr>
<tr>
<td>“neoplasm”</td>
<td>“tumor”</td>
</tr>
<tr>
<td>“Doctor”</td>
<td>“Dr.”</td>
</tr>
<tr>
<td>“Barack Obama”</td>
<td>“Obama”</td>
</tr>
<tr>
<td>“Jeffrey Jacob Abrams”</td>
<td>“J. J. Abrams”</td>
</tr>
<tr>
<td>“teacher”</td>
<td>“educator”</td>
</tr>
<tr>
<td>“1980”</td>
<td>“~30”</td>
</tr>
<tr>
<td>“3 mi”</td>
<td>“4.8 km”</td>
</tr>
<tr>
<td>“Hinton” - “DNNresearch” - “Google”</td>
<td>“Hinton” - “Google”</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Schema-less Graph Querying (SLQ)

Query

Prof., 70 yrs.

Toronto  Google

A Match

Geoffrey Hinton (1947-)

Univ. of Toronto  DNNResearch  Google

✓ Acronym transformation: ‘UT’ → ‘University of Toronto’
✓ Abbreviation transformation: ‘Prof.’ → ‘Professor’
✓ Numeric transformation: ‘~70’ → ‘1947’
✓ Structural transformation: an edge → a path
## Transformations

<table>
<thead>
<tr>
<th>Transformation</th>
<th>Category</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>First/Last token</td>
<td><strong>String</strong></td>
<td>“Barack Obama” &gt; “Obama”</td>
</tr>
<tr>
<td>Abbreviation</td>
<td><strong>String</strong></td>
<td>“Jeffrey Jacob Abrams” &gt; “J. J. Abrams”</td>
</tr>
<tr>
<td>Prefix</td>
<td><strong>String</strong></td>
<td>“Doctor” &gt; “Dr”</td>
</tr>
<tr>
<td>Acronym</td>
<td><strong>String</strong></td>
<td>&quot;International Business Machines&quot; &gt; &quot;IBM&quot;</td>
</tr>
<tr>
<td>Synonym</td>
<td><strong>Semantic</strong></td>
<td>“tumor&quot; &gt; “neoplasm&quot;</td>
</tr>
<tr>
<td>Ontology</td>
<td><strong>Semantic</strong></td>
<td>&quot;teacher&quot; &gt; &quot;educator&quot;</td>
</tr>
<tr>
<td>Range</td>
<td><strong>Numeric</strong></td>
<td>“~30” &gt; “1980”</td>
</tr>
<tr>
<td>Unit Conversion</td>
<td><strong>Numeric</strong></td>
<td>&quot;3 mi&quot; &gt; &quot;4.8 km&quot;</td>
</tr>
<tr>
<td>Distance</td>
<td><strong>Topology</strong></td>
<td>&quot;Pine&quot; - &quot;M:I&quot; &gt; &quot;Pine&quot; - &quot;J.J. Abrams&quot; - &quot;M:I&quot;</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Candidate Match Ranking

Query: $Q$

Candidate Match: $\varphi(Q)$

- **Features**
  - Node matching features: $F_V(v, \varphi(v)) = \sum_i \alpha_i f_i(v, \varphi(v))$
  - Edge matching features: $F_E(e, \varphi(e)) = \sum_j \beta_j g_j(e, \varphi(e))$

- **Overall Matching Score**

$$P(\varphi(Q) \mid Q) \propto \exp(\sum_{v \in V_Q} F_V(v, \varphi(v)) + \sum_{e \in E_Q} F_E(e, \varphi(e)))$$

Conditional Random Field
Query-specific Ranking via Relevance Feedback

• Generic ranking: sub-optimal for specific queries
  • By “Washington”, user A means Washington D.C., while user B might mean University of Washington

• Query-specific ranking: tailored for each query
  • But need additional query-specific information for further disambiguation

Relevance Feedback:
Users indicate the (ir)relevance of a handful of answers
Problem Definition

$Q$: A graph query
$G$: A knowledge graph
$\phi(Q)$: A candidate match to $Q$
$F(\phi(Q) | Q, \theta)$: A generic ranking function
$M^+$: A set of positive/relevant matches of $Q$
$M^-$: A set of negative/non-relevant matches of $Q$

*Graph Relevance Feedback (GRF):* Generate a query-specific ranking function $\tilde{F}$ for $Q$ based on $M^+$ and $M^-$
A General GRF Framework

<table>
<thead>
<tr>
<th>Input</th>
<th>Component</th>
<th>Stage Outcome</th>
<th>Final Ranking Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>$F(\phi(Q)</td>
<td>Q, \theta)$</td>
<td>Query-specific Tuning</td>
<td>$F(\phi(Q)</td>
</tr>
<tr>
<td>$\mathcal{M}^+, \mathcal{M}^-$</td>
<td>Type Inference</td>
<td>$Q^*$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Context Inference</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Query-specific Tuning

• The $\theta$ represents (query-independent) feature weights. However, each query carries its own view of feature importance.

• Find query-specific $\theta^*$ that better aligned with the query using user feedback.

$$g(\theta^*) = (1 - \lambda)\left(\sum_{\phi(Q)\in M^+} F(\phi(Q)|Q,\theta^*) - \frac{\sum_{\phi(Q)\in M^+} F(\phi(Q)|Q,\theta^*)}{|M^+|} - \frac{\sum_{\phi(Q)\in M^-} F(\phi(Q)|Q,\theta^*)}{|M^-|}\right) + \lambda R(\theta, \theta^*)$$

User Feedback

Regularization

[Su et al. KDD’15]
Type Inference

• Infer the implicit type of each query node
• The types of the positive entities constitute a composite type for each query node

[Su et al. KDD’15]
Context Inference

- **Entity context**: neighborhood of the entity
- The contexts of the positive entities constitute a composite context for each query node

[Su et al. KDD’15]
Experiment Setup

- Knowledge graph: DBpedia (4.6M nodes, 100M edges)
- Graph query sets: WIKI and YAGO

YAGO Class

- Naval Battles of World War II Involving the United States

Instances

- Battle of Midway
- Battle of the Caribbean

Graph Query

- Structured Information need

Answer

- Links between YAGO and DBpedia

[Su et al. KDD'15]
Evaluation with Explicit Feedback

- Explicit feedback: User gives relevance feedback on top-10 results
- GRF improves SLQ for over 100%
- Three GRF components complement each other

Metric: mean average precision (MAP)
Evaluation with Pseudo Feedback

• Pseudo feedback: Blindly assume top-10 results are correct
• Erroneous feedback information but no additional user effort

<table>
<thead>
<tr>
<th>MAP@K</th>
<th>1</th>
<th>5</th>
<th>10</th>
<th>20</th>
<th>50</th>
<th>100</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLQ_WIKI</td>
<td>0.23</td>
<td>0.21</td>
<td>0.24</td>
<td>0.25</td>
<td>0.27</td>
<td>0.28</td>
</tr>
<tr>
<td>GRF_WIKI</td>
<td>0.73</td>
<td>0.58</td>
<td>0.52</td>
<td>0.50</td>
<td>0.49</td>
<td>0.49</td>
</tr>
<tr>
<td>SLQ_YAGO</td>
<td>0.40</td>
<td>0.35</td>
<td>0.33</td>
<td>0.32</td>
<td>0.36</td>
<td>0.39</td>
</tr>
<tr>
<td>GRF_YAGO</td>
<td>0.82</td>
<td>0.66</td>
<td>0.60</td>
<td>0.57</td>
<td>0.58</td>
<td>0.61</td>
</tr>
</tbody>
</table>
Natural Language Query (a.k.a., Knowledge Based Question Answering)

Who is Justin Bieber’s sister?

Jazmyn Bieber

$\lambda x. \text{SiblingOf}(\text{justin_bieber}, x) \land \text{Gender}(x, \text{female})$

Figure credit to Scott Yih
Challenges

• Language mismatch
  • Lots of ways to ask the same question
    • Find terrorist organizations involved in September 11 attacks
    • Who did September 11 attacks?
    • The nine eleven were carried out with the involvement of what terrorist organizations?
  • All need to be mapped to the KB relation: terrorist_attack
Challenges

• Language mismatch

• Large search space
  • United_States has over 1 million neighbors in Freebase
Challenges

- Language mismatch
- Large search space
  - United_States has over 1 million neighbors in Freebase
- Scalability
  - How to scale up to more advanced inputs, and scale out to more domains?
  - KBQA data is highly domain-specific
Challenges

• Language mismatch

• Large search space
  • United_States has over 1 million neighbors in Freebase

• Scalability
  • How to scale up to more advanced inputs, and scale out to more domains?
  • KBQA data is highly domain-specific

• Compositionality
  • If a model understands relation A and B, can it answer A+B?
What will be covered

• Model
  • General pipeline
  • Semantic matching: CNN and Seq2Seq

• Data
  • Low-cost data collection via crowdsourcing
  • Cross-domain semantic parsing via neural transfer learning
General pipeline

Topic Entity Linking

Candidate Logical Form Generation

Semantic Matching

Execution

*Seq2Seq:*
[Jia and Liang, ACL’16]
[Liang et al. ACL’17]

*CNN:*
[Yih et al. ACL’15]
*Seq2Seq:*
[Su and Yan, EMNLP’17]
Query Graph

Who first voiced Meg on Family Guy?

\[ \lambda x. \exists y. \text{cast}(\text{FamilyGuy}, y) \land \text{actor}(y, x) \land \text{character}(y, \text{MegGriffin}) \]
Topic Entity Linking

- An advanced entity linker for short text

- Prepare surface form lexicon for KB entities
- Entity mention candidates: all consecutive word sequences in lexicon
- Score entity mention candidates with the statistical model, keep top-10 entities

[Yih et al. ACL’15]
Candidate Logical Form Generation

• (Roughly) enumerate all admissible logical forms up to a certain complexity (2-hop)
Semantic Matching (w/ CNN)

Discriminative model: 
\[ p(R|P) = \frac{\exp(cos(y_R, y_P))}{\sum_{R'} \exp(cos(y_{R'}, y_P))} \]

who voiced meg on <e>  cast-actor
Semantic Matching (w/ Seq2Seq)

Generative model: \( p(R|P) = \prod_i p(R_i|P, R_{<i}) \)

Encoder

Who | voiced | Meg | on | <e>

Decoder

Cast | actor | ... | ...
What will be covered

• Model
  • General pipeline
  • Semantic matching: CNN and Seq2Seq

• Data
  • Low-cost data collection via crowdsourcing
  • Cross-domain semantic parsing via neural transfer learning
Scalability

• Vertical scalability
  • Scale up to more complex inputs and logical constructs

Who was the head coach when Michael Jordan started playing for the Chicago Bulls?

In which season did Michael Jordan get the most points?

What team did Michael Jordan play for?
Scalability

• Vertical scalability
  • Scale up to more complex inputs and logical constructs

• Horizontal scalability
  • Scale out to more domains
  • Weather, calendar, hotel, flight, restaurant, ...
  • Knowledge base, relational database, API, robots, ...
  • Graph, table, text, image, audio, ...

• More **data** + Better (more data-efficient) model

On Generating Characteristic-rich Question Sets for QA Evaluation (EMNLP’16)
Cross-domain Semantic Parsing via Paraphrasing (EMNLP’17)
Building Natural Language Interfaces to Web APIs (CIKM’17)
Low-cost Data Collection via Crowdsourcing

1: Logical form generation

- \( \text{count}(\lambda x. \text{children}(\text{Eddard_Stark}, x) \land \text{place_of_birth}(x, \text{Winterfell})) \)

2: Canonical utterance generation

- "What is the number of person who is born in Winterfell, and who is child of Eddard Stark?"

3: Paraphrasing via crowdsourcing

- "How many children of Eddard Stark were born in Winterfell?"

[Wang+ ACL’15, Su+ EMNLP’16, Su+ CIKM’17]
Existing KBQA datasets mainly contain *simple questions*

“Where was Obama born?”

“What party did Clay establish?”

“What kind of money to take to bahamas?”

… …
GraphQuestions: A New KBQA Dataset with Rich Characteristics

- Structural complexity
  - “people who are on a gluten-free diet can’t eat what cereal grain that is used to make challah?”

- Quantitative analysis (functions)
  - “In which month does the average rainfall of New York City exceed 86 mm?”

- Commonness
  - “Where was Obama born?” vs.
  - “What is the tilt of axis of Polestar?”

- Paraphrase
  - “What is the nutritional composition of coca-cola?”
  - “What is the supplement information for coca-cola?”
  - “What kind of nutrient does coke have?”

- …

https://github.com/ysu1989/GraphQuestions
Large Room to Improve on GraphQuestions

<table>
<thead>
<tr>
<th>Model</th>
<th>Average F1 (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sempre (Berant+ EMNLP’13)</td>
<td>10.8</td>
</tr>
<tr>
<td>Jacana (Yao+ ACL’14)</td>
<td>5.1</td>
</tr>
<tr>
<td>ParaSempre (Berant+ ACL’14)</td>
<td>12.8</td>
</tr>
<tr>
<td>UDepLambda (Reddy+ EMNLP’17)</td>
<td>17.6</td>
</tr>
<tr>
<td>Para4QA (Li+ EMNLP’17)</td>
<td>20.4</td>
</tr>
</tbody>
</table>
Crowdsourcing is great, but…

- There is an unlimited number of application domains; prohibitive cost to collect (sufficient) training data for every one.

- **Transfer learning**: Use existing data of some source domains to help target domain

- **Problem**: KBQA data is highly domain-specific
What is **transferrable** in semantic parsing?

---

**In which season did Kobe Bryant play for the Lakers?**

\[ R[\text{season}].(\text{player.KobeBryant} \land \text{team.Lakers}) \]

\[ p(\text{team}, \text{"play for"}) \]

---

**When did Alice start working for Mckinsey?**

\[ R[\text{start}].(\text{employee.Alice} \land \text{employer.Mckinsey}) \]
• First convert logical forms to canonical utterances
• Train a neural paraphrase model on the source domains; adapt the model to the target domain
Why it works?

- Source domain: “play for” ⇒ “whose team is”
- Word embedding: “play” ⇒ “work”, “team” ⇒ “employer”
- Target domain: “work for” ⇒ “whose employer is”
Neural Transfer Learning for Semantic Parsing

Source Domain

Target Domain

Pre-trained Word Embedding
• Overnight dataset: 8 domains (basketball, calendar, etc.), each with a knowledge base
• For each target domain, use other 7 domains as source